DOCKET NO. 98-MET-069C1 (STMI01-01012) **PATENT**
Customer No. 30425

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of	:	David L. Isaman
Serial No.	:	09/443,160
Filed	:	November 19, 1999
For	:	SYMBOLIC STORE-LOAD BYPASS
Group No.	:	2183
Examiner	:	Idriss N. Alrobaye
Confirmation No.	:	6854

MAIL STOP AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

AMENDMENT AND RESPONSE TO FINAL OFFICE ACTION

This communication responds to a Final Office Action mailed July 10, 2009, which has a shortened statutory period for reply set to expire on October 10, 2009. As this Reply is being filed within two months of the mailing date of the Final Office Action, the Applicant requests an Advisory Action.

IN THE CLAIMS:

The current claims follow. For claims not marked as amended in this response, any difference in the claims below and the previous state of the claims is unintentional and in the nature of a typographical error.

1. (Canceled).

2. (Currently Amended) A pipelined microprocessor detecting a first instruction using first base and offset address values to load data from a first memory location that was previously stored to, wherein the first instruction is detected based upon the first base and offset address values and without computing a memory address equaling the first base address value added to the offset address ~~values~~ value in detecting the first instruction.

3. (Previously Presented) A pipelined microprocessor as claimed in claim 2 wherein the pipelined microprocessor detects a second instruction using second base and offset address values to store data into a second memory location that was previously read from, wherein the second instruction is detected based upon the second base and offset address values and without computing a memory address equaling the second base address value added to the offset address values in detecting the second instruction.

4.-5. (Canceled).

6. (Previously Presented) A pipelined microprocessor as claimed in claim 2 wherein the pipelined microprocessor examines base and offset address values used to access memory locations by store instructions that store data into the memory locations, and detects load instructions that load data from memory locations corresponding to base and offset address values identical to the base and offset address values used by the store instructions.

7. (Previously Presented) A pipelined microprocessor as claimed in claim 3 wherein the pipelined microprocessor examines base and offset address values used to access memory locations by load instructions that load data from the memory locations, and detects store instructions that store data into memory locations corresponding to base and offset address values identical to the base and offset address values used by the load instructions.

8. (Previously Presented) A pipelined microprocessor as claimed in claim 6 wherein the pipelined microprocessor detects identical offset address values and identical base address values in at least one register within the pipelined microprocessor.

9. (Previously Presented) A pipelined microprocessor as claimed in claim 7 wherein the pipelined microprocessor detects identical offset address values and identical base address values in at least one register within the pipelined microprocessor.

10. (Previously Presented) A pipelined microprocessor as claimed in claim 6 wherein the pipelined microprocessor comprises:

an instruction decode stage detecting load instructions that load data from memory locations corresponding to offset address values from an identical and base address values identical to offset address values and base address values used by prior store instructions that store data into the memory locations; and

a bypass element sending a bypass signal to an instruction execution stage of the pipelined microprocessor that indicates that a load instruction uses a base address value and an offset address value identical to a base address value and an offset address value used by a prior store instruction.

11. (Previously Presented) A pipelined microprocessor as claimed in claim 7 wherein the pipelined microprocessor comprises:

an instruction decode stage detecting store instructions that store data into memory locations using offset address values and base address values identical to offset address values and base address values used by prior load instructions that load data from memory locations; and

a bypass element sending a bypass signal to an instruction execution stage of the pipelined microprocessor that indicates that a store instruction uses a base address value and an offset address value identical to a base address value and an offset address value used by a prior load instruction.

12. (Currently Amended) A method for operating a pipelined microprocessor, comprising:

detecting, in the pipelined microprocessor, a first instruction using first base and offset address values to load data from a first memory location that was previously stored to, wherein the first instruction is detected based upon the first base and offset address values and without computing a memory address equaling the first base address value added to the offset address ~~values~~ value in detecting the first instruction.

13. (Previously Presented) A method for operating a pipelined microprocessor as claimed in claim 12, further comprising:

detecting, in the pipelined microprocessor, a second instruction using second base and offset address values to store data into a second memory location that was previously read from, wherein the second instruction is detected based upon the second base and offset address values and without computing a memory address equaling the second base address value added to the offset address values in detecting the second instruction.

14.-15. (Canceled).

16. (Previously Presented) A method for operating a pipelined microprocessor as claimed in claim 12, further comprising:

examining, in the pipelined microprocessor, base and offset address values used to access memory locations by store instructions that store data into the memory locations; and

detecting load instructions that load data from memory locations corresponding to base and offset address values identical to the base and offset address values used by the store instructions.

17. (Previously Presented) A method for operating a pipelined microprocessor as claimed in claim 13, further comprising:

examining, in the pipelined microprocessor, base and offset address values used to access memory locations by load instructions that load data from memory locations; and

detecting said instructions that store data into memory locations corresponding to base and offset address values identical to the base and offset address values used by the load instructions.

18. (Previously Presented) A method for operating a pipelined microprocessor as claimed in claim 16, further comprising:

detecting, in an instruction decode stage of the pipelined microprocessor, load instructions that load data from memory locations corresponding to offset address values and base address values identical to offset address values and base address values used by prior store instructions that store data into the memory locations; and

sending a bypass signal from a bypass element to an instruction execution stage of the pipelined microprocessor, wherein the bypass signal indicates that a load instruction uses a base address value and an offset address value identical to a base address value and an offset address value used by a prior store instruction.

19. (Previously Presented) A method for operating a pipelined microprocessor as claimed in claim 17, further comprising:

detecting, in an instruction decode stage of the pipelined microprocessor, store instructions that store data into memory locations using offset address values and base address values identical to offset address values and base address values used by prior load instructions that load data from memory locations; and

sending a bypass signal from a bypass element to an instruction execution stage of the pipelined microprocessor, wherein the bypass signal indicates that a load instruction uses a base address value and an offset address value identical to a base address value and an offset address value used by a prior store instruction.

20. (Currently Amended) A method for operating a pipelined microprocessor, comprising:

detecting a first instruction that stores data to a first memory location, the first instruction comprising syntax for computing an effective address for the first memory location;

detecting a second instruction that loads data from a second memory location, the second instruction comprising syntax for computing an effective address for said second memory location;

determining the syntax for the first instruction and the syntax for the second instruction;

using the syntax for the first instruction and the syntax for the second instruction to determine a relationship between the first memory location and the second memory location, without using the effective address of the first memory location or the effective address of the second memory location to determine the relationship between the first memory location [[nd]] and the second memory location; and

using the relationship to determine whether to perform one of the first instruction and the second instruction.

21. (Previously Presented) A method for operating a pipelined microprocessor as claimed in claim 20, wherein the syntax for the first instruction and the syntax for the second instruction refer to an identical memory location.

REMARKS

Claims 2, 3, 6-13 and 16-21 are pending in the application.

Claims 2, 3, 6-13 and 16-21 have been rejected.

Claims 2, 12 and 20 have been amended as set forth herein.

Claims 2, 3, 6-13 and 16-21 remain pending in this application.

Reconsideration of the claims is respectfully requested. The Applicant makes the aforementioned amendments and subsequent arguments to place this application in condition for allowance. Alternatively, the Applicant makes these amendments and offers these arguments to properly frame the issues for appeal.

I. CLAIM OBJECTIONS

Claim 20 was objected to because of minor informalities and has been amended to correct these informalities. The Applicant respectfully requests that the Objections to Claim 20 be withdrawn.

II. CLAIM REJECTIONS -- 35 U.S.C. § 112

Claims 2-3, 6-13 and 16-19 were rejected under 35 U.S.C. § 112, first paragraph as claiming subject matter that is not described in the specification in a manner enabling one skilled in the relevant art to make or use the claimed invention. This rejection is respectfully traversed.

Any analysis of whether a particular claim is supported by the disclosure in an application requires a determination of whether that disclosure, when filed, contained sufficient information

regarding the subject matter of the claims as to enable one skilled in the pertinent art to make and use the claimed invention. MPEP § 2164.01, p. 2100-193 (8th ed., rev. 4, October 2005). The test of enablement is whether one reasonably skilled in the art could make or use the invention from the disclosures in the patent coupled with information known in the art without undue experimentation. *Id.* A patent need not teach, and preferably omits, what is well known in the art. *Id.* The Patent Office has the initial burden of establishing a reasonable basis to question the enablement provided for the claimed invention. MPEP § 2164.04 at 2100-197. The minimal requirement for a proper enablement rejection is to give reasons for the uncertainty of the enablement. *Id.*

The Office Action contends that "without computing a memory address equaling the first base address value added to the offset address value in detecting the first instruction." The Office Action argues that the specification shows that **after** the first instruction has been detected it does not generate the actual effective address. However, the contention by the Office Action is unsupported in the specification as the specification clearly states:

> The invention provides a method and system for operating a pipelined microprocessor more quickly, by detecting instructions without having to actually compute the referenced external memory address. (See Specification page 4, lines 3-6)
>
> The instruction decode stage 120 parses the instructions 151 to determine what types of instructions 151 they are (such as instructions 151 that load data from external memory or store data to external memory). As part of parsing the instructions 151, and **in addition to determine what operations** the instructions 151 command the microprocessor to perform, the instruction decode stage 120 determines the syntax of any address in external memory that the instructions 151 refer to as operands. (See Specification page 6, lines 17-22) (Emphasis added)

Accordingly, the specification as filed describes the subject matter of the elements on which the rejection is based. Therefore, the Applicant respectfully requests that the Examiner withdraw the § 112 rejection.

III. CLAIM REJECTIONS -- 35 U.S.C. § 102

Claims 2, 12 and 20 were rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 6,216,200 to Yeager (hereinafter "*Yeager*"). Claims 2-3, 6-13 and 16-21 were rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,666,506 to Hesson (hereinafter "*Hesson*"). These rejections are respectfully traversed.

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. MPEP § 2131, p. 2100-76 (8th ed., rev. 4, October 2005) (*citing In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990)). Anticipation is only shown where each and every limitation of the claimed invention is found in a single prior art reference. *Id.* (*citing Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 U.S.P.Q.2d 1051, 1053 (Fed. Cir. 1987)).

The Office Action contends that *Yeager* teaches each and every element as recited and arranged in independent Claim 2. The Office Action argues that *Yeager* (col. 30, lines 43-49, "comparison of virtual addresses") teaches wherein the first instruction is detected based upon the first base and offset address values. The Office Action also argues that *Yeager* (col. 30, lines 43-49;

"comparison of virtual addresses" to see if there's store-to-load dependency) teaches without computing a memory address equaling the first base address value added to the offset address value in detecting the first instruction. The Office Action states that "[t]here is no memory address computation equaling the first base address value added to the offset address values, thus reads on the limitation; see also abstract wherein the dependencies is [sic] detected before virtual address calculation."

Yeager teaches comparing virtual addresses that are, for indexed address calculations, formed by "base+index". (*Yeager*, col. 9, lines 21-22). *Yeager* expressly teaches that "dependencies" may be tracked before actual calculation of the virtual address based on a "presumption of the dependency" and such dependency is dynamically corrected **once the address becomes available**. (*Yeager*, Abstract) The dependency is a relationship between an instruction and the operands produced by a prior instruction. (See generally, *Yeager*, col. 1, lines 29-33). Accordingly, *Yeager* does not teach or suggest "detecting a first instruction using first base and offset address values to load data from a first memory location that was previously stored to, wherein the first instruction is detected based upon the first base and offset address values and without using a memory address equaling to the first address value added to the offset address value."

Further, *Hesson* teaches using the virtual addresses – that is, the effective addresses, as opposed to the physical or "real" addresses – of memory locations to determine correspondence of two memory locations. The Office Action contends that *Hesson* (col. 4, line 64 – col. 5, line 13)

teaches that the virtual address includes a base and effective address (Office Action, page 6). The cited portion of *Hesson* states:

> The store barrier cache 11 performs a comparison of the next instruction prefetch fetch buffer virtual address against the virtual instruction address field in each of its cache entries. In general, the store barrier hit is defined as a match of the next instruction virtual address with the store barrier cache virtual address field and the condition that the store barrier bit is asserted. There may be more than one store barrier hit within the instruction fetch buffer specified by the next instruction prefetch buffer virtual address. Therefore, the store barrier cache output that is produced is the first store barrier cache hit within the store barrier cache line that is greater than or equal to the next instruction prefetch buffer address. If a store barrier cache hit results from the next instruction prefetch buffer virtual address, then the store barrier cache hit control output is a logic one. If no match is found, then the store barrier cache hit control output is a logic zero. (*Hesson*, col. 4, line 64 – col. 5, line 13).

The cited portion of *Hesson* contains no disclosure that teaches or suggest that the virtual address has a base and effective address. This contention by the Office Action that "the virtual address has a base and effective address" is unsupported in *Hesson*. Accordingly, *Hesson* fails to teach or suggest "detecting a first instruction using first base and offset address values to load data from a first memory location that was previously stored to, wherein the first instruction is detected based upon the first base and offset address values and without using a memory address equaling to the first address value added to the offset address value."

Claim 20 recites using the syntax for the first instruction and the syntax for the second instruction to determine a relationship between the first memory location and the second memory location, without using the effective address for the first memory location or the effective address for

the second memory location. Such a feature is not found in the cited references. The Office Action contends that the "virtual addresses are equivalent to the claimed memory addresses." As stated hereinabove, *Yeager* expressly teaches that the virtual addresses are used to correct the presumed dependencies." (*Yeager*, Abstract). Therefore, *Yeager* cannot reasonably be interpreted as teaching "using the syntax for the first instruction and the syntax for the second instruction to determine a relationship between the first memory location and the second memory location, without using the effective address for the first memory location or the effective address for the second memory location."

Hesson teaches using the virtual addresses – that is, the effective addresses, as opposed to the physical or "real" addresses – of memory locations to determine correspondence of two memory locations. The interpretation of "the virtual address has a base and effective address" is unsupported in *Hesson*. No basis for such as limitation, other than to contrive a basis for rejection of the claim, exists. Therefore, *Hesson* cannot reasonably be interpreted as teaching "using the syntax for the first instruction and the syntax for the second instruction to determine a relationship between the first memory location and the second memory location, without using the effective address for the first memory location or the effective address for the second memory location."

Accordingly, the Applicant respectfully requests that the Examiner withdraw the § 102 rejection with respect to these claims.

CONCLUSION

As a result of the foregoing, the Applicant asserts that the remaining Claims in the Application are in condition for allowance, and respectfully requests an early allowance of such Claims.

If any issues arise, or if the Examiner has any suggestions for expediting allowance of this Application, the Applicant respectfully invites the Examiner to contact the undersigned at the telephone number indicated below or at ***jmockler@munckcarter.com***.

The Commissioner is hereby authorized to charge any additional fees connected with this communication or credit any overpayment to Deposit Account No. 50-0208.

Respectfully submitted,

MUNCK CARTER, LLP

Date: September 10, 2009

John T. Mockler
Registration No. 39,775

P.O. Box 802432
Dallas, Texas 75380
(972) 628-3600 (main number)
(972) 628-3616 (fax)
E-mail: *jmockler@munckcarter.com*